IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:	§	Group Art Unit: 2166
	§	
Ajay Kumar	§	Examiner: Wong, Joseph D.
Bala Dutt	§	
Venugopal Rao K	§	
Sankara R. Bhogi	§	
Srinivasan Kannan	§	Atty. Dkt. No.: 5681-15200
	§	
Serial No. 10/618,810	§	
	§	
	§	
Filed: July 14, 2003	§	
	§	
Title:	§	
	§	
Read/Write Lock Transaction Manager Freezing	§	
	§	

APPEAL BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir/Madam:

Further to the Notice of Appeal filed July 28, 2009, Appellants present this Appeal Brief. Appellants respectfully request that the Board of Patent Appeals and Interferences consider this appeal.

I. REAL PARTY IN INTEREST

The subject application is owned by Sun Microsystems, Inc., a corporation organized and existing under and by virtue of the laws of the State of Delaware, and now having its principal place of business at 4150 Network Circle, Santa Clara, CA 95054.

II. RELATED APPEALS AND INTERFERENCES

No other appeals, interferences or judicial proceedings are known which would be related to, directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

Claims 1, 3-11 and 13-58 are pending and stand finally rejected. Claims 2 and 12 have been canceled. The rejection of claims 1, 3-11 and 13-58 is being appealed. A copy of the appealed claims, as currently pending, is included in the Claims Appendix herein below.

IV. STATUS OF AMENDMENTS

No amendments have been submitted subsequent to the final rejection.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Claim 1 is directed toward a system that includes one or more processors and a memory coupled to the one or more processors and configured to store program instructions (see e.g., Figure 10, items 1010-1020; page 20, line 6 – page 21, line 12). The program instructions are executable by the one or more processors to implement one or more applications configured to initiate one or more atomic transactions (see e.g., applications 100 of Figure1, page 8, line 1 – page 9, line 7); each of the one or more atomic transactions comprises requests to access one or more data sources (see e.g., page 11, lines 1-15). The program instructions are configured to implement a transaction manager configured to control state changes of the one or more atomic transactions initiated by the one or more applications (see e.g., transaction manager 605 of Figure 6, page 13, line 26 – page 14, line 11). For each given atomic transaction, the transaction manager is configured to request permission to change the state of the given atomic transaction (see e.g., page 3, lines 17-29; page 12, lines 22-30). The program instructions are configured to implement a transaction freeze manager configured to pause the transaction manager in response to a pause request by withholding said permission to change the state of the given atomic transaction (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30). The transaction manager is configured to not change the state of the given atomic transaction without said permission (see e.g., Figure 5, items 510-540; page 13 lines 1 – 24). The transaction freeze manager is configured to resume the transaction manager in response to a resume request by granting said permission to change the state of the given atomic transaction (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30).

Claim 10 is directed toward a system that includes one or more processors and a memory coupled to the one or more processors and configured to store program instructions (see e.g., Figure 10, items 1010-1020; page 20, line 6 – page 21, line 12). The program instructions are executable by the one or more processors to implement one or more applications configured to initiate one or more atomic transactions (see e.g., applications 100 of Figure1, page 8, line 1 – page 9, line7); each of the one or more

atomic transactions comprises requests to access one or more data sources (see e.g., page 11, lines 1-15). The program instructions are configured to implement one or more transaction managers configured to control state changes of the one or more atomic transactions initiated by the one or more applications (see e.g., transaction manager 605 of Figure 6, page 13, line 26 – page 14, line 11). For each given atomic transaction, the transaction manager is configured to request permission to change the state of the given atomic transaction (see e.g., page 3, lines 17-29; page 12, lines 22-30). The program instructions are configured to implement one or more transaction freeze managers configured to pause the transaction manager in response to a pause request by withholding said permission to change the state of the given atomic transaction (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30). The one or more transaction manager are configured to not change the state of the given atomic transaction without said permission (see e.g., Figure 5, items 510-540; page 13 lines 1 – 24). The one or more transaction freeze managers are configured to resume the transaction manager in response to a resume request by granting said permission to change the state of the given atomic transaction (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30).

Claim 11 is directed toward a system that includes one or more processors and a memory coupled to the one or more processors and configured to store program instructions (see e.g., Figure 10, items 1010-1020; page 20, line 6 – page 21, line 12). The program instructions are executable by the one or more processors to implement one or more applications configured to initiate one or more atomic transactions (see e.g., applications 100 of Figure1, page 8, line 1 – page 9, line7); each of the one or more atomic transactions comprises requests to access one or more data sources (see e.g., page 11, lines 1-15). The program instructions are configured to implement a transaction manager configured to control state changes of the one or more atomic transactions initiated by the one or more applications (see e.g., transaction manager 605 of Figure 6, page 13, line 26 – page 14, line 11). For each given atomic transaction, the transaction manager is configured to request a read lock on a stored transaction freeze object to change the state of the given atomic transaction (see e.g., page 16, lines 12-21; Figure 8,

items 810 – 870). The program instructions are configured to implement a transaction freeze manager configured to pause the transaction manager in response to a pause request by withholding said read lock for said stored transaction freeze object (see e.g., page 15, line 2 - page 16, line 2; Figure 8, items 810-870). The transaction manager is configured to not change the state of the given atomic transaction without said read lock (see e.g., Figure 5, items 510-540; page 13 lines 1 – 24; see e.g., page 15, line 2 - page 16, line 2). The transaction freeze manager is configured to resume the transaction manager in response to a resume request by granting said read lock for said stored transaction freeze object (see e.g., Figure 8, items 810-870; page 16, lines 4-21).

Claim 20 is directed toward a system that includes one or more processors and a memory coupled to the one or more processors and configured to store program instructions (see e.g., Figure 10, items 1010-1020; page 20, line 6 – page 21, line 12). The program instructions are executable by the one or more processors to implement one or more applications configured to initiate one or more atomic transactions (see e.g., applications 100 of Figure1, page 8, line 1 – page 9, line7); each of the one or more atomic transactions comprises requests to access one or more data sources (see e.g., page 11, lines 1-15). The program instructions are configured to implement one or more transaction managers configured to control state changes of the one or more atomic transactions initiated by the one or more applications (see e.g., transaction manager 605 of Figure 6, page 13, line 26 – page 14, line 11). For each given atomic transaction, the transaction manager is configured to request a read lock on a stored transaction freeze object to change the state of the given atomic transaction (see e.g., page 16, lines 12-21; Figure 8, items 810 – 870). The program instructions are configured to implement one or more transaction freeze managers configured to pause the transaction manager in response to a pause request by withholding said read lock for said stored transaction freeze object (see e.g., page 15, line 2 - page 16, line 2; Figure 8, items 810-870). The one or more transaction managers are configured to not change the state of the given atomic transaction without said read lock (see e.g., Figure 5, items 510-540; page 13 lines 1 – 24; see e.g., page 15, line 2 - page 16, line 2). The one or more transaction freeze managers are configured to resume the transaction manager in response to a resume

request by granting said read lock for said stored transaction freeze object (see e.g., Figure 8, items 810-870; page 16, lines 4-21).

Claim 21 is directed to a method that includes using one or more computers to perform receiving a pause request (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30). The method may also include using the one or more computers to perform pausing a transaction manager in response to the pause request by withholding permission to change the state of one or more transactions managed by the transaction manager (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30). The method may also include using the one or more computers to perform receiving a plurality of resume requests (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30). The method may also include using the one or more computers to perform resuming the transaction manager in response to the resume request by granting permission to change the state of the one or more transactions managed by the transaction manager (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30).

Claim 30 is directed toward a method that includes using one or more computers to perform receiving a pause request (see e.g., page 15, line 2 - page 16, line 2; Figure 8, items 810-870). The method may also include using the one or more computers to perform pausing a transaction manager in response to the pause request by withholding read locks on a stored transaction freeze object that identifies a respective atomic transaction (see e.g., page 15, line 2 - page 16, line 2; Figure 8, items 810-870). The method may also include using the one or more computers to perform receiving a resume request (see e.g., Figure 8, items 810-870; page 16, lines 4-21). The method may also include using the one or more computers to perform resuming the transaction manager in response to the resume request by granting read locks on the stored transaction freeze object that identifies the respective atomic transaction (see e.g., Figure 8, items 810-870; page 16, lines 4-21).

Claim 39 is directed toward a computer readable storage medium storing program instructions that are computer-executable to receive a pause request (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30). The program instructions are also configured to pause a transaction manager in response to the pause request by withholding permission to change the state of one or more transactions managed by the transaction manager (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30). The program instructions are also configured to receive a resume request (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30). The program instructions are also configured to resume the transaction manager in response to the resume request by granting permission to change the state of the one or more transactions managed by the transaction manager (see e.g., items 710-750 of Figure 7; page 14, line 13 – page 16, line 2; page 12, lines 22-30).

Claim 48 is directed to a computer readable storage medium storing program instructions that are computer-executable to receive a pause request (see e.g., page 15, line 2 - page 16, line 2; Figure 8, items 810-870). The program instructions are also configured to pause a transaction manager in response to the pause request by withholding read locks on a stored transaction freeze object that identifies a respective atomic transaction (see e.g., page 15, line 2 - page 16, line 2; Figure 8, items 810-870). The program instructions are also configured to receive a resume request (see e.g., Figure 8, items 810-870; page 16, lines 4-21). The program instructions are also configured to resume the transaction manager in response to the resume request by granting read locks on the stored transaction freeze object that identifies the respective atomic transaction (see e.g., Figure 8, items 810-870; page 16, lines 4-21).

The summary above describes various examples and embodiments of the claimed subject matter; however, the claims are not necessarily limited to any of these examples and embodiments. The claims should be interpreted based on the wording of the respective claims.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Claims 1, 10, 11, 20, 21, 30, 39, and 48 stand provisionally rejected for obviousness-type double patenting.

2. Claims 21-25, 27 and 29 stand finally rejected under 35 U.S.C. § 102(e) as being anticipated by Oeltjen et al. (U.S. Publication 2004/0225972) (hereinafter "Oeltjen").

3. Claims 39-41, 43, 45 and 47 stand finally rejected under 35 U.S.C. § 102(e) as being anticipated by Ault et al. (U.S. Patent 6,237,019) (hereinafter "Ault").

4 Claims 1, 3-11 and 13-20 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Hagersten et al. (U.S. Patent 5,983,326) (hereinafter "Hagersten") in view of Fowler et al. (U.S. Patent 4,502,116) (hereinafter "Fowler").

5. Claim 26 stands finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Oeltjen in view of Hagersten.

6. Claim 28 stands finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Oeltjen in view of U.S. Patent 6,659,992.

7. Claims 30-32, 34, 36, 48-50, 52 and 54 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Ault in view of Oliver (U.S. Patent 6,029,190).

8. Claims 33, 38, 42, 51 and 56 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Ault in view of Oliver in further view of Hagersten.

9. Claims 35, 44 and 53 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Ault in view of Oliver, and in further view of Hagersten.

10. Claims 57-58 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Hagersten in view of Fowler and in further view of Oeltjen.

11. Claims 37, 46 and 55 stand finally rejected under Ault in view of Oliver and in further view of U.S. Patent 6,549,941.

VII. ARGUMENT

First Ground of Rejection:

Claims 1, 10, 11, 20, 21, 30, 39, and 48 stand provisionally rejected for obviousness-type double patenting. Appellants traverse this rejection for at least the following reasons.

The Examiner rejected claims 1, 10, 11, 20, 21, 30, 39, and 48 under the judiciary created doctrine of obviousness-type double patenting as being unpatentable over claims 1-36 of co-pending Application No. 10/618,828. Applicants traverse this rejection on the grounds that the Examiner has not stated a proper *prima facie* rejection. The only support given by the Examiner for the rejection is:

> In claim 1, the instant application adds, "wherein for each transaction". Other synonymous phraseology or obvious variations are present such as instant claim 1 recites "not change the state of the transaction without said permission" and the copending claim 1 recites "does not allow the one or more transactions to complete."

However, according to MPEP 804.II.B.1, "the analysis employed in an obviousness-type double patenting determination parallels the guidelines for a 35 U.S.C. 103(a) rejection." This section of the MPEP also states that the same "factual inquires … that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are employed when making an obviousness-type double patenting analysis." MPEP 804.II.B.1 also states that the Examiner should list the differences between **each** rejected claim and the claims of the other patent/application, and for **each** difference the Examiner should give the reasons why a person of ordinary skill in the art would conclude that the invention defined in the claim is an obvious variation of the invention defined in a claim of the other patent/application. The Examiner has not specifically addressed **each difference** of **each claim** of the present application compared to the claims of the other applications. For instance, claim 20 of the present application recites "a transaction freeze manager configured to pause the transaction manager in

response to a pause request by withholding said read lock for said stored transaction freeze object." However, claims 1-36 of co-pending Application No. 10/618,828 do not appear to mention a read lock, a stored transaction freeze object, much less a transaction freeze manager configured to pause the transaction manager in response to a pause request by withholding said read lock for said stored transaction freeze object. The Examiner clearly has not met the requirements stated in MPEP 804.II.B.1 to establish a *prima facie* obviousness-type double patenting rejection. Accordingly, Applicants respectfully request removal of the double patenting rejection of claims 1, 10, 11, 20, 21, 30, 39, and 48.

Second Ground of Rejection:

Claims 21-25, 27 and 29 stand finally rejected under 35 U.S.C. § 102(e) as being anticipated by Oeltjen. Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claim 21- 24, 27 and 29

I. In regard to claim 21, Oeltjen fails to teach receiving a pause request, and pausing a transaction manager in response to the pause request by withholding permission to change the state of one or more transactions managed by the transaction manager.

The Examiner cites paragraphs 9 and 38 of Oeltjen. Oeltjen describes an automated framework and methodology for the development, testing, validation, and documentation of the design of semiconductor products that culminates in the release of a design kit having a flow manager and flow file to actualize a methodology to design a semiconductor product (Abstract). The Examiner equates the "errors result[ing] from tools failing" of paragraph 9 of Oeltjen to the "pausing a transaction manager in response to the pause request by withholding permission to change the state of one or more transactions managed by the transaction manager" limitation of Applicants' claim. The Examiner asserts that "such an error necessarily blocks a pause or restart."

First, irrespective of the correctness of the Examiner's assertion, the cited art has nothing to do with pausing a transaction manager, much less pausing a transaction manager "by withholding permission to change the state of one or more transactions managed by the transaction manager." There Oeltjen reference is completely silent with respect this particular manner in which to pause a transaction manager.

Second, the "pausing a transaction manager" limitation of Applicants' claim is performed "in response to the pause request." Under the Examiner's interpretation of the

reference, Oeltjen would have to teach that the "error" was performed in response to a request for an error. Clearly, the "error[s]" taught by Oeltjen are unwanted errors (hence the term "error"); nowhere does the reference teach requesting such errors. Such a strained interpretation of the reference is clearly improper. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to address this argument.**

Third, even were one to somehow consider the "errors" of Oeltjen to be pause requests (which is clearly improper), Oeltjen fails to teach performing errors in response to requests for errors. Accordingly, Oeltjen cannot teach pausing a transaction manager in response to a pause request by withholding permission to change the state of one or more transactions managed by the transaction manager. **In the response to arguments section of the Final Office Action mailed April 28, 2009,** the Examiner asserts "pausing is responsive to the pause command but there is nothing in the claim that requires the withholding of the permission to be exclusively bound to the pause command." The Examiner is incorrect. The "withholding of the permission" referred to by the Examiner is a specific manner in which the pausing of the transaction manager is performed. The plain language of Appellants' claim explicitly recites "pausing the transaction manager in response to a pause request by withholding permission to change the state of one or more transactions managed by the transaction manager." It is improper for the Examiner to ignore the plain language of Appellants' claim.

II. Furthermore, the cited art fails to teach or suggest receiving a plurality of resume requests, and resuming the transaction manager in response to the resume request by granting permission to change the state of the one or more transactions managed by the transaction manager. The Examiner cites paragraph 38, in particular the portion of that describes "commands that permit stepping through and editing code." Stepping through and editing code has nothing to do with resuming a transaction manager, much less resuming a transaction manager "by granting permission to change the state of the one or more transactions managed by the transaction manager" as recited in Applicants claim. Even a cursory review of Oeltjen would lead one of ordinary skill in

the art to conclude that the limitations of Applicants' claim are not disclosed by the reference. Since Oeltjen fails to teach all of the limitations of Applicants' claim, Oeltjen cannot be said to anticipate Applicants' claim.

In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner asserts "claim 21 recites the argued limitation behind a preface of 'configured to' which means that the claim does not necessarily do the step but needs to be capable of doing the steps." The Examiner is incorrect. Claim 21 does not recite limitations behind the preface "configured to." Instead, claim 21 recites a specific manner of pausing and resuming a transaction manger that includes the positive claim limitations "pausing a transaction manager in response to the pause request by withholding permission to change the state of one or more transactions managed by the transaction manager" and "resuming the transaction manager in response to the resume request by granting permission to change the state of the one or more transactions managed by the transaction manager." It is improper for the Examiner to ignore the plain language of Appellants' claim.

For at least the reasons presented above, the rejection of claim 21 is unsupported by the cited art and removal thereof is respectfully requested.

Claims 25

In regard to claim 25, Oeltjen fails to teach wherein the transaction freeze manager is configured to queue received state transition permission requests and transaction manager pause requests in the order received. The Examiner cites Figure 7 "where queuing is broadly interpreted to include any depth of buffering even 1 because all operations within the real world require time and are not instantaneously (sic), see [35]." The "buffering" to which the Examiner cites has nothing to do with state transition requests nor transaction manager pause requests. Furthermore, the claim explicitly recites multiple ones of such requests; accordingly, the Examiner's reliance on "any

depth of buffering even 1" is not commensurate with the specific limitations of Applicant's claim.

For at least the reasons presented above, the rejection of claim 25 is unsupported by the cited art and removal thereof is respectfully requested.

Third Ground of Rejection:

Claims 39-41, 43, 45 and 47 stand finally rejected under 35 U.S.C. § 102(e) as being anticipated by Ault et al. (U.S. Patent 6,237,019) (hereinafter "Ault"). Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claim 39- 41, 43 and 47

I. In regard to claim 39, Ault fails to teach program instruction configured to receive a pause request, and pause a transaction manager in response to the pause request by withholding permission to change the state of one or more transactions managed by the transaction manager. The Examiner equates the suspension of a calling thread (column 2, lines 48-62) of Ault to be equivalent to the limitation "paus[ing] a transaction manager in response to the pause request *by withholding permission to change the state of one or more transactions managed by the transaction manager*" as recited in Applicants claim.

First, suspending a thread as taught by Ault is not the same as "withholding permission to change the state of one or more transactions managed by the transaction manager" as recited in Applicants claim. Nor does suspending a thread inherently include "withholding permission to change the state of one or more transactions managed by the transaction manager." In fact, the Examiner fails to cite any portion of the reference that he considers to be equivalent to the claimed "permission." Applicants note the present claim does not merely recite "pausing a transaction manager." Instead, Applicants claim recites pausing a transaction manager in a specific manner "by withholding permission to change the state of one or more transactions managed by the transaction manager." **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

Second, Applicants claim recites "paus[ing] a transaction manager *in response to the pause request.*" Even were one to consider the suspension of Ault's thread to be equivalent to the pausing of a transaction manager (which is clearly improper), such thread suspension is not performed in response to a request to suspend the thread. Accordingly, even under the Examiner's reasoning, the cited art cannot teach "paus[ing] a transaction manager in response to the pause request by withholding permission to change the state of one or more transactions managed by the transaction manager" as recited in Applicants claim. Since the cited art fails to teach this specific limitation of Applicants' claim, the cited art cannot be said to anticipate Applicants' claim. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

II. The cited art fail to teach program instructions configured to receive a resume request, and resume the transaction manager in response to the resume request by granting permission to change the state of the one or more transactions managed by the transaction manager. The Examiner cites the granting of a semaphore as illustrated in Figure 3. Granting a semaphore for a resource as taught by Ault is not the same as resuming the transaction manager in response to the resume request *by granting permission to change the state of the one or more transactions managed by the transaction manager*. Nor does granting a semaphore for a resource inherently include resum[ing] the transaction manager in response to the resume request *by granting permission to change the state of the one or more transactions managed by the transaction manager* as recited in Applicants' claim. Anyone of ordinary skill in the art would recognize that a "resource" can be accessed without changing the state of a transaction. Since Ault fails to teach the aforesaid limitations of Applicants' claim and since the limitations of Applicants' claim are not inherently included in any of the teachings of Ault, Ault cannot be said to anticipate Applicants' claim. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

For at least the reasons presented above, the rejection of claim 39 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 45

In regard to claim 45, the cited art fails to teach wherein the transaction freeze manager is configured to grant a state transition permission request if the transaction manager is not paused. The Examiner cites Figures 2-3 and 5A-5E. While the cited portions generally teach the management of a semaphore for access to a shared resource, the cited portions of Ault do not specifically teach the granting of a state transition permission request if a transaction manager is not paused. The Examiner asserts that this limitation is a conditional limitation (presumably to overcome the deficiencies of the cited reference). However, the limitations of claim 45 are positive claims limitations that specify a particular manner in which a transaction freeze manager is configured. As such, these limitations carry patentable weight in the claim. Furthermore, Ault fails to teach a transaction freeze manager that grants a state transition permission request if a transaction manager is not paused. In fact, the Examiner has failed to specify the particular element on which he is relying to teach the transaction manager, much less a transaction freeze manager configured to perform specific actions if such a transaction manager is not paused.

For at least the reasons presented above, the rejection of claim 45 is unsupported by the cited art and removal thereof is respectfully requested.

Fourth Ground of Rejection:

Claims 1, 3-11 and 13-20 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Hagersten in view of Fowler. Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claims 1, 3- 6, 9 and 10

I. In regard to claim 1, the cited art fails to teach a transaction freeze manager configured to (i) pause the transaction manager in response to a pause request by withholding said permission to change the state of the given atomic transaction, and (ii) resume the transaction manager in response to a resume request by granting said permission to change the state of the given atomic transaction.

The Examiner acknowledges that Hagersten alone fails to teach this limitation of Applicants claim (*see e.g.,* Final Office Action mailed April 28, 2009; rejection of claim 1). Moreover, the Examiner relies on the teachings of Fowler in combination with Hagersten's "blocking unit" and/or "spin lock" to teach the aforesaid limitations. The Examiner also cites column 5, lines 40-53 of Fowler, which describes a "control section 80" that "provides user with toggle switches to configure the pause/resume synchronization of the multiprocessor system," and column 8, lines 52-56, which describes "a generated resume output request signal for concurrently providing a resumption control signal to each of the other processors concurrently to effect a resumption of program execution thereby." **While the cited art generally refers to "pausing" and "resuming" processor operations, the cited art clearly fails to teach pausing or resuming a transaction manager, much less pausing or resuming a transaction manager in the specific manner recited by Applicants' claim.** For instance, the cited art clearly fails to teach "a transaction freeze manager configured to pause the transaction manager in response to a pause request *by withholding said permission to change the state of the given atomic transaction*" as recited in Applicants'

claim. Fowler's "control system 80" and/or its "toggle switches" are not configured to withhold a permission to change the state of a given atomic transaction, much less pause a transaction manager in this manner. The cited art also fails to teach a transaction freeze manager configured to resume the transaction manager in response to a resume request "*by granting said permission to change the state of the given atomic transaction*" as recited in Applicants' claim. The "resume output request signal" (Fowler; column 8, lines 52-56; cited by Examiner) is not configured to grant permission to change the state of a given atomic transaction, much less resume a transaction manager in this manner. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

II. Furthermore, the cited art fails to teach or suggest a transaction manager configured to control state changes of the one or more atomic transactions initiated by the one or more applications, wherein for each given atomic transaction, the transaction manager is configured to request permission to change the state of the given atomic transaction. It appears the Examiner is relying on the "home agent" and/or the "transaction blocking unit" of Hagersten to teach the claimed transaction manager and the "transfer of data from a source to a destination" (Hagersten; column 8, lines 20-24) to teach the transactions of Applicants' claim. (*see e.g.,* Final Office Action mailed April 28, 2009; pages 13-14).

First, the cited art does not teach that the "transfer of data from a source to a destination" is actually an atomic transaction. The only atomic operations described by Hagersten are atomic test-and-set operations (Hagersten; column 2, line 57 – column 3, line 4) that "allow[] [a] process to determine whether a lock bit associated with the memory region is cleared and to atomically set the bit." According to Hagersten, the state of such "test-and-set operations" are controlled *by the individual processes*, not the "home agent" (nor the "transaction blocking unit"). Furthermore, such "test-and-set operations" occur *before* the "transfer of data from a source to a destination" described in column 8, lines 20-24. **In the response to arguments section of the Final Office**

Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.

Secondly, the cited art does not teach or suggest that the "home agent" or the "transaction blocking unit" control state changes associated with the "transfer of data from a source to a destination." In fact, as described above, the only atomic operations disclosed by the cited art are atomic "test-and-set operations," which are controlled by individual processes not the "home agent" nor the "transaction blocking unit." The cited art fails to explicitly teach state changes of the "transfer of data from a source to a destination," much less that the "home agent" or "transaction blocking unit" are configured to control such state changes. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

Thirdly, the cited art fails to teach a transaction manager configured to *request permission* to change the state of a given atomic transaction. The "request" on which the Examiner relies (*see e.g.,* receive request 162 of Figure 7) is not issued by the "home agent" nor is such request issued by the "transaction block unit" (on which the Examiner relies to teach the transaction manager). Instead, the "request" on which the Examiner relies is actually a coherency request received by the home agent, not a request issued by the home agent (nor a request issued by the transaction blocking unit) (see e.g., Hagersten, column 21, lines 26-27). Fowler, even when considered in combination with Hagersten, fails to overcome the above-described deficiencies. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

III. Even were the teachings of Fowler to somehow be combined with Hagersten, such combination would not result in Appellants' claimed invention. For instance, the "request" on which the Examiner relies (*see e.g.,* receive request 162 of Figure 7) to teach the claimed request for "permission to change the state of a given atomic transaction" (issued by a transaction manager) is not issued by the "home agent" nor is

such request issued by the "transaction block unit" (on which the Examiner relies to teach the transaction manager). Accordingly, even were one to somehow combine the teachings of the cited art in the manner suggested by the Examiner, such combination would not result in the specific system of claim 1 that includes a transaction manager configured to control state changes of the one or more atomic transactions initiated by the one or more applications, wherein for each given atomic transaction, the transaction manager is configured to request permission to change the state of the given atomic transaction.

For at least the reasons presented above, the rejection of claim 1 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 7

In regard to claim 7, the cited art fails to teach or suggest wherein the transaction freeze manager is configured to grant the said permission request in response to determining that the transaction manager is not paused. The Examiner cites "Fig. 7, 'not blocked'-> 'set blocked status'" of Hagersten. Hagersten describes this portion of his method in column 21, lines 42-59, which is reproduced below.

> Conversely, home agent 102 transitions to a check state 168 upon receipt of a coherent request. Check state 168 is used to detect if coherency activity is in progress for the coherency unit affected by the coherency request. If the coherency activity is in progress (i.e. the coherency information is blocked), then home agent 102 remains in check state 168 until the in-progress coherency activity completes. Home agent 102 subsequently transitions to a set state 170.
>
> During set state 170, home agent 102 sets the status of the directory entry storing the coherency information corresponding to the affected coherency unit to blocked. The blocked status prevents subsequent activity to the affected coherency unit from proceeding, simplifying the coherency protocol of computer system 10. Depending upon the read or write nature of the transaction corresponding to the received coherency request, home agent 102 transitions to read state 172 or write reply state 174.

The cited portion of Hagersten pertains to the home agent (on which the Examiner relies to teach the transaction manager) evaluating a block status (e.g., check block status 168). However, the cited art does not teach a transaction freeze manager that is configured to grant a permission request in response to determining that the home agent is not paused. Instead, the cited art teaches "detect[ing] if coherency activity is in progress for the coherency unit" but does not teach "determining that the transaction manager is not paused" much less teach a transaction freeze manager that is configured to grant the said permission request in response to determining that the transaction manager is not paused, as recited in claim 7.

For at least the reasons presented above, the rejection of claim 7 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 8

In regard to claim 8, the cited art fails to teach or suggest wherein the transaction freeze manager is configured to grant the pause request in response to determining that the transaction manager is not paused and there are no outstanding state transition permission requests received prior to the pause request. The Examiner cites "Fig. 7, 'not blocked'-> 'set blocked status'" of Hagersten. Hagersten describes this portion of his method in column 21, lines 42-59, which is reproduced above. The cited portion of Hagersten pertains to the home agent (on which the Examiner relies to teach the transaction manager) evaluating a block status (e.g., check block status 168). However, the cited art does not teach a transaction freeze manager that is configured to grant the pause request in response to determining that the transaction manager is not paused and there are no outstanding state transition permission requests received prior to the pause request. Instead, the cited art teaches "detect[ing] if coherency activity is in progress for the coherency unit" but does not teach "determining that the transaction manager is not paused" much less determining that there are "no outstanding state transition permission requests received prior to the pause request," as

recited in claim 8. Nor does the cited art teach a transaction freeze manager configured to grant a pause request in response to such a determination.

For at least the reasons presented above, the rejection of claim 8 is unsupported by the cited art and removal thereof is respectfully requested.

Claims 11, 13- 16, 19 and 20

I. In regard to claim 11, the cited art fails to teach a transaction freeze manager configured to (i) pause the transaction manager in response to a pause request by withholding said read lock for said stored transaction freeze object, and (ii) resume the transaction manager in response to a resume request by granting said read lock for said stored transaction freeze object.

The Examiner acknowledges that Hagersten alone fails to teach this limitation of Applicants claim. Moreover, the Examiner relies on the teachings of Fowler in combination with Hagersten's "blocking unit" and/or "spin lock" to teach the aforesaid limitations. The Examiner also cites column 5, lines 40-53 of Fowler, which describes a "control section 80" that "provides user with toggle switches to configure the pause/resume synchronization of the multiprocessor system," and column 8, lines 52-56, which describes "a generated resume output request signal for concurrently providing a resumption control signal to each of the other processors concurrently to effect a resumption of program execution thereby." **While the cited art generally refers to "pausing" and "resuming" processor operations, the cited art clearly fails to teach pausing or resuming a transaction manager, much less pausing or resuming a transaction manager in the specific manner recited by Applicants' claim.** For instance, the cited art clearly fails to teach "a transaction freeze manager configured to pause the transaction manager in response to a pause request *by withholding said read lock for said stored transaction freeze object*" as recited in Applicants' claim. Fowler's "control system 80" and/or its "toggle switches" are not configured to withhold a read lock for said stored transaction freeze object, much less pause a transaction manager in

this manner. **In fact, the cited art fails to teach or suggest anything about a transaction freeze object, much less read locks on transaction freeze objects.** Nor does the Examiner specifically cite any portion of the reference that he considers to be equivalent to the freeze object. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

The cited art also fails to teach a transaction freeze manager configured to resume the transaction manager in response to a resume request "*by granting said read lock for said stored transaction freeze object*" as recited in Applicants' claim. The "resume output request signal" (Fowler; column 8, lines 52-56; cited by Examiner) is not configured to grant a read lock for a stored transaction freeze object, much less resume a transaction manager in this manner. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

II. Furthermore, the cited art fails to teach or suggest a transaction manager configured to control state changes of the one or more atomic transactions initiated by the one or more applications, wherein for each given atomic transaction, the transaction manager is configured to request a read lock on a stored transaction freeze object to change the state of the given atomic transaction. It appears the Examiner is relying on the "home agent" and/or the "request agent" of Hagersten to teach the claimed transaction manager and the "transfer of data from a source to a destination" (Hagersten; column 8, lines 20-24) to teach the transactions of Applicants' claim. (*see e.g.,* Final Office Action mailed April 28, 2009; pages 18-19).

First, the cited art does not teach that the "transfer of data from a source to a destination" is actually an atomic transaction. The only atomic operations described by Hagersten are atomic test-and-set operations (Hagersten; column 2, line 57 – column 3, line 4) that "allow[] [a] process to determine whether a lock bit associated with the memory region is cleared and to atomically set the bit." According to Hagersten, the

state of such "test-and-set operations" are controlled *by the individual processes*, not the "home agent" (nor the "request agent"). Furthermore, such "test-and-set operations" occur *before* the "transfer of data from a source to a destination" described in column 8, lines 20-24. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

Secondly, the cited art does not teach or suggest that the "home agent" or the "request agent" control state changes associated with the "transfer of data from a source to a destination." In fact, as described above, the only atomic operations disclosed by the cited art are atomic "test-and-set operations," which are controlled by individual processes not the "home agent" nor the "transaction blocking unit." The cited art fails to explicitly teach state changes of the "transfer of data from a source to a destination," much less that the "home agent" or "transaction blocking unit" are configured to control such state changes. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

Thirdly, the cited art fails to teach a transaction manager configured to *request a read lock on a stored transaction freeze object to change the state of the given atomic transaction*. The "request" on which the Examiner presumably relies (*see e.g.,* receive request 162 of Figure 7) is not issued by the "home agent." Instead, the "request" on which the Examiner relies is actually a coherency request received by the home agent, not a request issued by the home agent (see e.g., Hagersten, column 21, lines 26-27). Fowler, even when considered in combination with Hagersten, fails to overcome the above-described deficiencies. **In the response to arguments section of the Final Office Action mailed April 28, 2009, the Examiner fails to provide a response to this argument.**

III. Even were the teachings of Fowler to somehow be combined with Hagersten, such a combination would not result in Appellants' claimed invention. For instance, since the only atomic operations taught by Fowler and Hagersten, whether considered singly or in combination, are atomic test test-and-set operations (Hagersten; column 2,

line 57 – column 3, line 4) controlled by individual processes, and not controlled by the "home agent" nor the "request agent" (on which the Examiner relies to teach transaction manager), combining the teaching of the cited art would not result in a transaction manager configured to control state changes of one or more atomic transactions.

For at least the reasons presented above, the rejection of claim 11 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 17

In regard to claim 17, the cited art fails to teach or suggest wherein the transaction freeze manager is configured to grant read locks in response to determining that the transaction manager is not paused. The Examiner cites "Fig. 7, 'not blocked'-> 'set blocked status'" of Hagersten. Hagersten describes this portion of his method in column 21, lines 42-59, which is reproduced above. The cited portion of Hagersten pertains to the home agent (on which the Examiner relies to teach the transaction manager) evaluating a block status (e.g., check block status 168). However, the cited art does not teach a transaction freeze manager that is configured to grant read locks in response to determining that the home agent is not paused. Instead, the cited art teaches "detect[ing] if coherency activity is in progress for the coherency unit" but does not teach "determining that the transaction manager is not paused" much less teach a transaction freeze manager that is configured to grant read locks in response to determining that the transaction manager is not paused, as recited in claim 17.

For at least the reasons presented above, the rejection of claim 17 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 18

In regard to claim 18, the cited art fails to teach or suggest wherein the transaction freeze manager is configured to grant a write lock in response to determining that the transaction manager is not paused and there are no outstanding read lock requests received prior to the write lock request. The Examiner cites "Fig. 7, 'not blocked'-> 'set blocked status'" of Hagersten. Hagersten describes this portion of his method in column 21, lines 42-59, which is reproduced above. The cited portion of Hagersten pertains to the home agent (on which the Examiner relies to teach the transaction manager) evaluating a block status (e.g., check block status 168). However, the cited art does not teach a transaction freeze manager that is configured to grant a write lock in response to determining that the transaction manager is not paused and there are no outstanding read lock requests received prior to the write lock request. Instead, the cited art teaches "detect[ing] if coherency activity is in progress for the coherency unit" but does not teach "determining that the transaction manager is not paused" much less determining that there are "no outstanding read lock request received prior to the write lock request," as recited in claim 18.

For at least the reasons presented above, the rejection of claim 18 is unsupported by the cited art and removal thereof is respectfully requested.

Fifth Ground of Rejection:

Claim 26 stands finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Oeltjen in view of Hagersten. Appellants traverse this rejection for at least the following reasons.

Claim 26

In regard to claim 26, Appellants assert the Examiner has failed to provide a proper reason as to why one of ordinary skill in the art would have combined the teachings of Oeltjen with the teachings of Hagersten. The Examiner asserts such a combination would have been obvious "because it provides for a multiprocessing system employing an enhanced blocking mechanism for read-to-share transactions as suggested in Hagersten," Col. 4, Lines 45-49. However, Oeltjen does not utilize "read-to-share transactions." Accordingly, one of ordinary skill in the art would not combine the teachings of the cited art to "provide for a multiprocessing system employing an enhanced blocking mechanism *for read-to-share transactions*." At best, even were one to attempt to combine the teachings of the cited art in the manner suggested by the Examine, such an attempt would result in a system where Oeltjen's "automated framework and methodology for the development and testing, validation, and document of the design of semiconductor products" (Oeltjen, Abstract) operates independently of Hagersten's "enhanced blocking mechanism" feature. "'[R]ejections on obviousness cannot be sustained by mere conclusory statements; instead, there *must* be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness'" (emphasis added). *KSR v. Teleflex*, 550 U.S. ___, 82 USPQ2d 1385, 1396. **Since the Examiner has not presented such reasoning, the Examiner's rejection is improper.**

Sixth Ground of Rejection:

Claim 28 stands finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Oeltjen in view of U.S. Patent 6,659,992. Appellants traverse this rejection for at least the following reasons.

Claim 28

In regard to claim 28, the rejection is improper because the reference cited is non-analogous art. The Examiner refers to U.S. Patent 6,659,992 as the "Armangau" reference. However, U.S. Patent 6,659,992 entitled "Absorbent article instanteously storing liquid in a predefined pattern" has no inventor by the name of "Armangau." Furthermore, U.S. Patent 6,659,992 pertains to an "absorbent article such as a diaper" (U.S. Patent 6,659,992, abstract). The cited art is non-analogous art that does not disclose the limitations of claim 28. Accordingly, the rejection is improper and the Examiner has failed to establish a *prima facie* rejection of claim 28.

Seventh Ground of Rejection:

Claims 30-32, 34, 36, 48-50, 52 and 54 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Ault in view of Oliver. Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claims 30-32, 34, 36, 48-50, 52 and 54

The cited art fails to teach pausing a transaction manager in response to the pause request by withholding read locks on a stored transaction freeze object that identifies a respective atomic transaction. The Examiner cites the teaching of Ault and Oliver, none of which teach the specific limitations of claim 30, whether such references are considered singly or in combination).

First, the cited art fails to teach anything about "read locks on a stored transaction freeze object that identifies a respective atomic transaction," much less "pausing a transaction manager in response to the pause request by withholding read locks on a stored transaction freeze object that identifies a respective atomic transaction" as recited in Applicants claim. Instead, Ault teaches semaphores for "resources"; nowhere does Ault teach or suggest that such resources are stored transaction freeze objects that identify respective transactions. Nor does the cited art teach withholding read locks on such objects to pause a transaction manager in response to a pause request, as recited in Applicants' claim. For reasons similar to those presented above, the cited art also fails to teach granting read locks on the stored transaction freeze object that identifies a respective atomic transaction. **Second**, the teachings of Oliver (on which the Examiner relies to teach "read locks") fail to overcome the deficiencies of Ault. For example, Oliver describes "protected resources" (abstract); however, nowhere does Oliver describe such resources as stored transaction freeze objects identifying respective transactions, much less withholding or granting read locks for such objects. **Accordingly, even were**

the teachings of the cited art combined, the resultant combination would not meet the specific limitations of Applicants' claim.

In the response to arguments section of Final Office Action mailed April 28, 2009, the Examiner appears to imply that limitations of Appellants' claim do not carry patentable weight because they are preceded by the word "by." More specifically, the Examiner asserts "the preposition 'by' can more broadly refer to the vicinity or coincident rather than direct causal relationship." Irrespective of how the word "by" may be used in isolation, the manner in which the word "by" is clear within the specific context provided by the plain language of Appellants claim. Within the specific context of plain language of the Appellants claim, it would be clear to one of ordinary skill in the art that the phrase "by withholding read locks on a stored transaction freeze object that identifies a respective atomic transaction" specifies the particular manner in which the "pausing a transaction manager" is performed. Not only does the Examiner's interpretation of the word "by" make no sense within the specific context of the plain language of Applicant's claim, such interpretation is also inconsistent with how one of ordinary skill in the art would interpret the plain language of Appellants' claim. Further, Appellants note that, aside from the remarks regarding the use of "by," the Examiner provided no response to Appellants argument that the cited art fails to teach the specific limitations of claim 30.

For at least the reasons presented above, the rejection of claim 30 is unsupported by the cited art and removal thereof is respectfully requested. Similar remarks apply to claim 48.

Eighth Ground of Rejection:

Claims 33, 38, 42, 51 and 56 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Ault in view of Oliver in further view of Hagersten.. Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claims 33 and 51

In regard to claim 33, the cited art fails to teach or suggest "wherein the transaction freeze manager is configured to receive requests for write locks on the stored transaction freeze object to pause the transaction manager from an administrative entity." The Examiner asserts that Ault does not expressly teach write locks and that Oliver teaches write locks. However, Applicant's claim does not merely recited "write locks"; instead, Applicant's claim recites "write locks on the stored transaction freeze object," which is not taught by the cited references. Nor does the cited art teach pausing a transaction manager in this way. Additionally, the Examiner relies on the teachings of Hagersten to teach the claimed "administrative entity." While Hagersten teaches an administrative request type, nowhere does Hagersten teach that the particular type of request for a write lock is received from such entity.

In regard to the Examiner's "intended use" remarks, Appellants note that the use of functional limitations to define an invention has been expressly approved by the courts. *See, e.g., In re Swinehart*, 439 F.2d 210, 169 USPQ 226 (CCPA 1971). In fact, the Court has held that a functional claim limitation can distinguish over the prior art "because it set definite boundaries on the patent protection sought." *In re Barr*, 444 F.2d 588, 170 USPQ 33 (CCPA 1971).

For at least the reasons presented above, the rejection of claim 33 is unsupported by the cited art and removal thereof is respectfully requested.

Claims 38 and 56

In regard to claim 38 and 51, the cited art fails to teach or suggest wherein the transaction freeze manager is configured to not grant locks if a write lock on the stored transaction freeze object is currently held by an administrative entity. The Examiner asserts that Ault does not expressly teach write locks and that Oliver teaches write locks. Additionally, the Examiner relies on the teachings of Hagersten to teach the claimed "administrative entity." However, Applicants claim does not merely recite "write locks" and an "administrative entity." Instead, claim 38 specifies a transaction freeze manager configured in a particular way, namely configured to not grant locks if a write lock on the stored transaction freeze object is currently held by an administrative entity. A transaction freeze manager configured in this particular manner is neither taught nor suggested by the cited references.

For at least the reasons presented above, the rejection of claim 38 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 42

Appellants assert the cited art fails to teach the specific limitations of claim 42 for at least the reasons presented above with respect to claim 40.

Ninth Ground of Rejection:

Claims 35, 44 and 53 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Ault in view of Oliver, and in further view of Hagersten. Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claims 35 and 53

In regard to claim 35, Appellants assert the Examiner has failed to provide a proper reason as to why one of ordinary skill in the art would have combined the teachings of Hagersten with Ault and Oliver. The Examiner asserts such a combination would have been obvious "because it provides for a multiprocessing system employing an enhanced blocking mechanism for read-to-share transactions as suggested in Hagersten," Col. 4, Lines 45-49. However, Oliver and Ault do not employ "read-to-share transactions." Accordingly, one of ordinary skill in the art would not combine the teachings of the cited art to "provide for a multiprocessing system employing an enhanced blocking mechanism *for read-to-share transactions*." "'[R]ejections on obviousness cannot be sustained by mere conclusory statements; instead, there *must* be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness'" (emphasis added). *KSR v. Teleflex*, 550 U.S. ___, 82 USPQ2d 1385, 1396. **Since the Examiner has not presented such reasoning, the Examiner's rejection is improper.**

For at least the reasons presented above, the rejection of claim 35 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 44

In regard to claim 44, Appellants assert the Examiner has failed to provide a proper reason as to why one of ordinary skill in the art would have combined the

teachings of Hagersten with Ault and Oliver. The Examiner asserts such a combination would have been obvious "because it provides for a multiprocessing system employing an enhanced blocking mechanism for read-to-share transactions as suggested in Hagersten," Col. 4, Lines 45-49. However, Oliver and Ault do not employ "read-to-share transactions." Accordingly, one of ordinary skill in the art would not combine the teachings of the cited art to "provide for a multiprocessing system employing an enhanced blocking mechanism *for read-to-share transactions.*"

For at least the reasons presented above, the rejection of claim 44 is unsupported by the cited art and removal thereof is respectfully requested.

Tenth Ground of Rejection:

Claims 57-58 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Hagersten in view of Fowler and in further view of Oeltjen. Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claim 57

In regard to claim 57, the cited art fails to teach that wherein the transaction manager is configured to, without said permission, perform one or more operations associated with a current state of the given atomic transaction. The Examiner relies on paragraph [0038] of Oeltjen, which is reproduced below:

> To fix or debug failed flows, resource developers 422 - 428 can take advantage of several debug features of the flow manager 460, specifically commands that permit stepping through and editing code such as step, next, run, continue, as well as pause, stop or kill a task. The flow manager 460 also gives a flow developer 424 or other user full control over the location within the flow to manually set the current flow step and/or to establish flow nesting.

The debug features of Oeltjen have nothing to do with an atomic transaction. Such features (e.g., next, run, continue, etc.) are commands for evaluating software code and are not operations associated with a current state of the given atomic transaction.

For at least the reasons presented above, the rejection of claim 57 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 58

In regard to claim 58, the cited art fails to teach that wherein the transaction manager is configured to, without said permission, perform one or more operations associated with a current state of the given atomic transaction. The Examiner relies on

paragraph [0038] of Oeltjen, which is reproduced above. However, the debug features described at paragraph [0038] of Oeltjen have nothing to do with an atomic transaction. Such features (e.g., next, run, continue, etc.) are commands for evaluating software code and are not operations associated with a current state of the given atomic transaction.

For at least the reasons presented above, the rejection of claim 57 is unsupported by the cited art and removal thereof is respectfully requested.

Eleventh Ground of Rejection:

Claims 37, 46 and 55 stand finally rejected under Ault in view of Oliver and in further view of U.S. Patent 6,549,941. Appellants traverse this rejection for at least the following reasons. Different groups of claims are addressed under their respective subheadings.

Claims 37, 46 and 55

In regard to claim 37, the rejection is improper because the reference cited is non-analogous art. The Examiner refers to U.S. Patent 6,549,941 as the "Armangau" reference. However, U.S. Patent 6,549,941 entitled "Software system and methods for resubmitting form data to related web sites" has no inventor by the name of "Armangau." Furthermore, U.S. Patent 6,549,941 pertains to "resubmitting form data" (U.S. Patent 6,549,941, abstract). The cited art is non-analogous art that does not disclose the limitations of claim 37. Accordingly, the rejection is improper and the Examiner has failed to establish a *prima facie* rejection of claim 37.

CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejection of claims 1, 3-11 and 13-58 was erroneous, and reversal of his decision is respectfully requested.

The Commissioner is authorized to charge the appeal brief fee and any other fees that may be due to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5681-15200/RCK.

Respectfully submitted,

/Robert C. Kowert/
Robert C. Kowert, Reg. #39,255
Attorney for Appellants

Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
(512) 853-8850

Date: October 28, 2009

VIII. CLAIMS APPENDIX

The claims on appeal are as follows.

1. A system, comprising:

one or more processors;

memory coupled to the one or more processors and configured to store program instructions executable by the one or more processors to implement:

one or more applications configured to initiate one or more atomic transactions, wherein each of the one or more atomic transactions comprises requests to access one or more data sources;

a transaction manager configured to control state changes of the one or more atomic transactions initiated by the one or more applications; wherein for each given atomic transaction, the transaction manager is configured to request permission to change the state of the given atomic transaction; and

a transaction freeze manager configured to pause the transaction manager in response to a pause request by withholding said permission to change the state of the given atomic transaction;

wherein the transaction manager is configured to not change the state of the given atomic transaction without said permission;

wherein the transaction freeze manager is configured to resume the transaction manager in response to a resume request by granting said permission to change the state of the given atomic transaction.

3. The system as recited in claim 1, wherein the transaction freeze manager is a part of the transaction manager.

4. The system as recited in claim 1, wherein the transaction freeze manager is configured to receive requests to pause the transaction manager from an administrative entity.

5. The system as recited in claim 1, wherein the transaction freeze manager is configured to queue received state transition permission requests and transaction manager pause requests in the order received.

6. The system as recited in claim 5, wherein the transaction freeze manager is configured to service queued state transition permission requests and transaction manager pause requests in FIFO order.

7. The system as recited in claim 1, wherein the transaction freeze manager is configured to grant the said permission request in response to determining that the transaction manager is not paused.

8. The system as recited in claim 1, wherein the transaction freeze manager is configured to grant the pause request in response to determining that the transaction manager is not paused and there are no outstanding state transition permission requests received prior to the pause request.

9. The system as recited in claim 1, wherein the transaction freeze manager is configured to not grant requests if the transaction manager is paused.

10. A system, comprising a plurality of computer systems coupled by one or more networks, wherein the plurality of computer systems comprise:

one or more processors; and

memory coupled to the one or more processors and configured to store program instructions executable by the one or more processors to implement one or more application servers comprising:

one or more applications configured to initiate one or more atomic transactions, wherein each of the one or more atomic transactions comprises requests to access one or more data sources;

one or more transaction managers configured to control state changes of the one or more atomic transactions initiated by the one or more applications; wherein for each given atomic transaction, the one or more transaction managers are configured to request permission to change the state of the given atomic transaction; and

one or more transaction freeze managers configured to pause the transaction manager in response to a pause request by withholding said permission to change the state of the given atomic transaction;

wherein the one or more transaction managers are configured to not change the state of the given atomic transaction without said permission;

wherein the one or more transaction freeze managers are configured to resume the transaction manager in response to a resume request by granting said permission to change the state of the given atomic transaction.

11. A system, comprising:

one or more processors;

memory coupled to the one or more processors and configured to store program instructions executable by the one or more processors to implement:

one or more applications configured to initiate one or more atomic transactions, wherein each of the one or more atomic transactions comprises requests to access one or more data sources;

a transaction manager configured to control state changes of the one or more atomic transactions initiated by the one or more applications; wherein for each given atomic transaction, the transaction manager is configured to request a read lock on a stored transaction freeze object to change the state of the given atomic transaction; and

a transaction freeze manager configured to pause the transaction manager in response to a pause request by withholding said read lock for said stored transaction freeze object;

wherein the transaction manager is configured to not change the state of the given atomic transaction without said read lock; and

wherein the transaction freeze manager is configured to resume the transaction manager in response to a resume request by granting said read lock for said stored transaction freeze object.

13. The system as recited in claim 11, wherein the transaction freeze manager is a part of the transaction manager.

14. The system as recited in claim 11, wherein the transaction freeze manager is configured to receive requests for write locks on the stored transaction freeze object from an administrative entity to pause the transaction manager.

15. The system as recited in claim 11, wherein the transaction freeze manager is configured to queue received lock requests in the order received.

16. The system as recited in claim 15, wherein the transaction freeze manager is configured to service queued lock requests in FIFO order.

17. The system as recited in claim 11, wherein the transaction freeze manager is configured to grant read locks in response to determining the transaction manager is not paused.

18. The system as recited in claim 11, wherein the transaction freeze manager is configured to grant a write lock in response to determining the transaction manager is not paused and there are no outstanding read lock requests received prior to the write lock request.

19. The system as recited in claim 11, wherein the transaction freeze manager is configured to not grant locks in response to determining a write lock on the stored transaction freeze object is currently held by an administrative entity.

20. A system, comprising a plurality of computer systems coupled by one or more networks, wherein the plurality of computer systems comprise:

one or more processors; and

memory coupled to the one or more processors and configured to store program instructions executable by the one or more processors to implement one or more application servers comprising:

one or more applications configured to initiate one or more atomic transactions, wherein each of the one or more atomic transactions comprises requests to access one or more data sources; and

one or more transaction managers configured to control state changes of the one or more atomic transactions initiated by the one or more applications; wherein for each given atomic transaction, the one or more transaction managers are configured to request a read lock on a stored transaction freeze object to change the state of the given atomic transaction; and

one or more transaction freeze managers configured to pause the transaction manager in response to a pause request by withholding said read lock for said stored transaction freeze object;

wherein the one or more transaction managers are configured to not change the state of the given atomic transaction without said read lock;

wherein the one or more transaction freeze managers are configured to resume the transaction manager in response to a resume request by granting said read lock for said stored transaction freeze object.

21. A method, comprising:

using one or more computers to perform:

receiving a pause request;

pausing a transaction manager in response to the pause request by withholding permission to change the state of one or more transactions managed by the transaction manager;

receiving a plurality of resume requests; and

resuming the transaction manager in response to the resume request by granting permission to change the state of the one or more transactions managed by the transaction manager.

22. The method as recited in claim 21, wherein a transaction freeze manager grants and withholds said permission.

23. The method as recited in claim 22, wherein the transaction freeze manager is a part of the transaction manager.

24. The method as recited in claim 22, wherein the transaction freeze manager is configured to receive requests to pause the transaction manager from an administrative entity.

25. The method as recited in claim 22, wherein the transaction freeze manager is configured to queue received state transition permission requests and transaction manager pause requests in the order received.

26. The method as recited in claim 25, wherein the transaction freeze manager is configured to service queued state transition permission requests and transaction manager pause requests in FIFO order.

27. The method as recited in claim 22, wherein the transaction freeze manager is configured to grant a state transition permission request if the transaction manager is not paused.

28. The method as recited in claim 22, wherein the transaction freeze manager is configured to grant a transaction manager pause request if the transaction manager is not paused and there are no outstanding state transition permission requests received prior to the pause request.

29. The method as recited in claim 22, wherein the transaction freeze manager is configured to not grant requests if the transaction manager is paused.

30. A method, comprising:

using one or more computers to perform:

receiving a pause request;

pausing a transaction manager in response to the pause request by withholding read locks on a stored transaction freeze object that identifies a respective atomic transaction;

receiving a resume request; and

resuming the transaction manager in response to the resume request by granting read locks on the stored transaction freeze object that identifies the respective atomic transaction.

31. The method as recited in claim 30, wherein a transaction freeze manager grants and withholds the read locks.

32. The method as recited in claim 31, wherein the transaction freeze manager is a part of the transaction manager.

33. The method as recited in claim 31, wherein the transaction freeze manager is configured to receive requests for write locks on the stored transaction freeze object to pause the transaction manager from an administrative entity.

34. The method as recited in claim 31, wherein the transaction freeze manager is configured to queue received lock requests in the order received.

35. The method as recited in claim 34, wherein the transaction freeze manager is configured to service queued lock requests in FIFO order.

36. The method as recited in claim 31, wherein the transaction freeze manager is configured to grant a read lock if the transaction manager is not paused.

37. The method as recited in claim 31, wherein the transaction freeze manager is configured to grant a write lock if the transaction manager is not paused, and there are no outstanding read lock requests received prior to the write lock request, and there are no outstanding read locks.

38. The method as recited in claim 31, wherein the transaction freeze manager is configured to not grant locks if a write lock on the stored transaction freeze object is currently held by an administrative entity.

39. A computer readable storage medium storing program instructions, wherein the program instructions are computer-executable to:

receive a pause request;

pause a transaction manager in response to the pause request by withholding permission to change the state of one or more transactions managed by the transaction manager;

receive a resume request; and

resume the transaction manager in response to the resume request by granting permission to change the state of the one or more transactions managed by the transaction manager.

40. The computer readable storage medium as recited in claim 39, wherein a transaction freeze manager grants and withholds said permission.

41. The computer readable storage medium as recited in claim 40, wherein the transaction freeze manager is a part of the transaction manager.

42. The computer readable storage medium as recited in claim 40, wherein the transaction freeze manager is configured to receive requests to pause the transaction manager from an administrative entity.

43. The computer readable storage medium as recited in claim 40, wherein the transaction freeze manager is configured to queue received state transition permission requests and transaction manager pause requests in the order received.

44. The computer readable storage medium as recited in claim 43, wherein the transaction freeze manager is configured to service queued state transition permission requests and transaction manager pause requests in FIFO order.

45. The computer readable storage medium as recited in claim 40, wherein the transaction freeze manager is configured to grant a state transition permission request if the transaction manager is not paused.

46. The computer readable storage medium as recited in claim 40, wherein the transaction freeze manager is configured to grant a transaction manager pause request if

the transaction manager is not paused and there are no outstanding state transition permission requests received prior to the pause request.

47. The computer readable storage medium as recited in claim 40, wherein the transaction freeze manager is configured to not grant requests if the transaction manager is paused.

48. A computer readable storage medium storing program instructions, wherein the program instructions are computer-executable to:

receive a pause request;

pause a transaction manager in response to the pause request by withholding read locks on a stored transaction freeze object that identifies a respective atomic transaction;

receive a resume request; and

resume the transaction manager in response to the resume request by granting read locks on the stored transaction freeze object that identifies the respective atomic transaction.

49. The computer readable storage medium as recited in claim 48, wherein a transaction freeze manager grants and withholds the read locks.

50. The computer readable storage medium as recited in claim 49, wherein the transaction freeze manager is a part of the transaction manager.

51. The computer readable storage medium as recited in claim 49, wherein the transaction freeze manager is configured to receive requests for write locks on the stored transaction freeze object to pause the transaction manager from an administrative entity.

52. The computer readable storage medium as recited in claim 49, wherein the transaction freeze manager is configured to queue received lock requests in the order received.

53. The computer readable storage medium as recited in claim 52, wherein the transaction freeze manager is configured to service queued lock requests in FIFO order.

54. The computer readable storage medium as recited in claim 49, wherein the transaction freeze manager is configured to grant a read lock if the transaction manager is not paused.

55. The computer readable storage medium as recited in claim 49, wherein the transaction freeze manager is configured to grant a write lock if the transaction manager is not paused, and there are no outstanding read lock requests received prior to the write lock request, and there are no outstanding read locks.

56. The computer readable storage medium as recited in claim 49, wherein the transaction freeze manager is configured to not grant locks if a write lock on the stored transaction freeze object is currently held by an administrative entity.

57. The system of claim 1, wherein the transaction manager is configured to, without said permission, perform one or more operations associated with a current state of the given atomic transaction.

58. The system of claim 11, wherein the transaction manager is configured to, without said read lock for said stored transaction freeze object, perform one or more operations associated with a current state of the given atomic transaction.

IX. EVIDENCE APPENDIX

No evidence submitted under 37 CFR §§ 1.130, 1.131 or 1.132 or otherwise entered by the Examiner is relied upon in this appeal.

X. RELATED PROCEEDINGS APPENDIX

There are no related proceedings.